**마감일: 9월 27일(09/27), 13:00**

# 개요

이 과제는 원형 링크 리스트 구현과 기본 폴리곤 클래스 구현의 두 부분으로 구성됩니다.

# 일반 노트

* *이 숙제 가이드라인을* 잘 읽어보세요. 지침을 따르지 않을 경우 코드의 작동 여부와 관계없이 0을 받을 수 있습니다.
* IDE(이클립스, IntelliJ IDEA 등)를 사용하지 마십시오.
  + 서브라임 텍스트(리눅스/맥/윈도우), 메모장++(윈도우) 또는 텍스트 랭글러(맥)를 추천합니다.
  + IDE는 종종 코드의 "패키지"를 만들어 자동 등급을 깨뜨립니다.
  + **패키지** 문제를 **해결하는 방법을 알고 있다면** 원하는 IDE를 사용할 수 있습니다. 그러나 우리는 이미 해결책을 추천했기 때문에 이 문제와 관련된 어떠한 질문에도 대답하지 않을 것입니다.
* 메서드 또는 클래스 서명을 변경하지 마십시오. 코드가 클래스 또는 메서드 이름 또는 서명을 변경하면 자동 0을 받게 됩니다.
* 코드가 컴파일되는지 확인합니다. 컴파일하지 않은 코드는 자동으로 0을 받습니다. 코드가 컴파일되지 않으면 잘못된 코드를 주석으로 달아서 더미 값(null 또는 -1과 같은)을 반환하여 나머지 코드를 컴파일할 수 있도록 하는 것이 더 나을 수 있습니다.
* 당신의 코드가 오토그래더에 의해 받아들여질 수 있도록 하기 위해서는 LearnUS에 코드를 제출하고 다시 다운로드한 후 다시 컴파일하여 제공된 테스트 스위트를 확인해야 합니다. 이렇게 하면 제출하는 파일이 정확하다는 것을 알 수 있습니다.

# 링크드 리스트

기본 연결 목록에는 여러 가지 변형이 있으며 실제로 자주 사용됩니다. 두 가지 일반적인 것은 이중으로 연결된 목록(각 노드에 다음 노드와 이전 노드에 대한 포인터가 있는 경우)과 순환 연결된 목록(링크된 목록의 꼬리가 *머리를 가리키는* 경우)입니다.

순환, 이중 연결된 목록을 구현합니다. 이 구현에는 헤드 포인터만 있습니다. 순환 연결 목록과 관련된 몇 가지 공통 작업을 다시 구현합니다. CircularLinkedList.java에서 구현해야 하는 메소드는 다음과 같습니다.

1. public CircularLinkedList(): Constructor method
2. public int size(): 링크된 목록의 요소 수를 반환합니다.
3. public boolean is Empty(): 연결된 목록에 요소가 없으면 True를 반환합니다. 그렇지 않으면 False를 반환합니다.
4. public TetHead(): 헤드가 *가리키는* 노드에 저장된 요소를 반환합니다.
5. public void rotate (Direction 방향): 링크된 목록을 회전하여 헤드가 주어진 방향으로 노드를 가리킵니다. 방향은 TO NEXT와 TO PREIVE의 두 상수가 있는 Enum입니다.
6. public void insert (T element): *헤드* 앞에 element를 삽입합니다. 즉, 새로운 노드가 *헤드의* 이전 노드가 됩니다.
7. public T delete(): 머리가 가리키는 요소를 삭제하고 다시 돌려줍니다. 목록이 비어 있으면 null을 반환합니다. 삭제 후 목록이 비어 있지 않으면 새 헤드가 *원래* 헤드의 다음 노드여야 합니다.
8. public boolean find( T target): 첫 번째 일치하는 요소를 찾고 해당 요소를 헤드 포인팅하도록 *합니다.* 일치하는 요소가 없는 경우, 머리가 가리키는 요소는 그대로 유지되어야 합니다. 검색이 성공하면 True를 반환하고 그렇지 않으면 False를 반환합니다. .equals() 메서드를 사용하여 두 항목이 동일한지 확인할 수 있습니다.

# 다각형

원형으로 연결된 목록의 일반적인 용도는 2차원 다각형의 경계를 나타내는 것입니다. 다각형의 경계는 원순으로 정렬된 정점 목록으로 표현할 수 있으므로 이전에 구현된 원순 연결 목록을 사용하여 다각형의 생성 및 조작을 처리할 수 있습니다.

원형 링크된 목록 위에 기본 다각형을 구현합니다. 순환 연결 목록 섹션에서 필요한 많은 메소드를 구현할 필요는 없지만(순환 연결 목록 구현에서 이미 수행했으므로) 특별한 알고리즘인 *point-in-polygon을 구현해야 합니다.*

* 1. **다각형의 점**

*다각형* P와 *점* P가 주어졌을 때, 한 가지 중요한 질문은 P의 *내부에* 존재하는지 *여부를 결정하는* 것입니다. 이 문제에 대한 인기 있는 알고리즘은 레이-캐스팅 알고리즘으로 알려져 있습니다. 알고리즘은 다음과 같습니다.

자료**:** *다각형* P ; 점 *p*

결과 **:** p가  *Pr* 안에 있는지 여부 *=* p에서 시작하는 광선이 *∞ 카운트 =* 0까지 확장되는 것

*for**edge e* **∈ pdo**

**만약에** *r이 교차합니다.*

*카운트* = *카운트* + 1

## 끝단

*카운트가* **홀수이면 True를** 반환합니다.

## 그 외의

반환 False

## 끝을

**IMT2000 3GPP -** Ray-Casting 알고리즘

여기서 모서리는 다각형 경계에서 두 개의 순차적 점으로 정의됩니다. 아래에서는 알고리즘의 시연을 볼 수 있습니다. 다각형 외부에 있는 점 *P* 1에서 연장되는 광선은 두 가장자리를 가로지르며 광선은

안쪽에 있는 P *2에서* 3을 가로지릅니다.



실제로 이 알고리즘에 대해 고려해야 할 몇 가지 에지 사례가 있지만( 테스트할 점이 다각형의 경계에 있는 점인 경우, 다각형에 3개의 점이 있지만 모두 직선인 경우 등), 이 과제에서는 테스트하지 않습니다. 테스트된 모든 다각형은 0이 아닌 영역(3개 이상의 점이 있는 경우)을 가지며 자체 교차하지 않습니다.

이 알고리즘의 경우 일반적으로 모든 광선이 작동하지만, 이 할당에서는 기울기 0의 광선(점 오른쪽으로 무한대로 확장됨을 의미함)을 사용해야 합니다. 이 기울기를 사용하면 두 가장자리가 만나는 지점에서 광선이 다각형과 교차하는 검정이 없음을 보장합니다. 다른 기울기를 사용하는 경우에는 이 에지 케이스를 고려해야 하므로 어려울 수 있습니다.

마지막으로, 폴리곤에 3개 미만의 점이 포함된 경우 이 알고리즘에 대해 항상 False를 반환해야 합니다.

# 일반 방향

* src/main 디렉토리의 파일 상단에 있는 댓글에 이름과 학생증 번호를 적어주세요.
* 필요한 모든 방법을 실행합니다.
* 파일에 아직 포함되지 않은 내용은 가져오면 안 됩니다.
* 필요한 반송 유형 및 에지 케이스에 주의해야 합니다.
* 우리가 제공한 모든 코드는 src/base 디렉토리에서 찾을 수 있습니다. 클래스/메소드가 정확히 무엇을 하는지 확실하지 않다면 코드를 참조하십시오.

# 제출절차

main.java를 제외한 src/main 디렉토리의 파일을 zip 파일로 제출합니다. 그래들 빌드 도구를 사용하여 제출할 zip 파일을 만들어야 합니다(컴플렉팅 섹션 참조).

이 할당의 경우 다음 세 개의 파일만 제출해야 합니다.

* 원형 연결 목록.java
* 폴리곤.자바
* 학생 ID 번호.txt

2023xxxxxxx의 이름을 변경해야 합니다.txt를 실제 학생 ID 번호로 입력합니다. 해당 텍스트 파일 안에 다음 텍스트를 포함해야 합니다. 마지막 마침표를 포함해서 아래의 모든 글을 꼭 써주시기 바랍니다.

*이 임무를 완수함에 있어서, 저는 어떠한 무단적인 도움도 주거나 받지 않았음을 맹세합니다.*

이 파일이 없으면 과제물에 0점을 받게 됩니다. 다른 텍스트 없이 학생증 이름을 정확히 *지어야* 합니다. 예를 들어 *2023123456.*txt는 맞지만 *2023123456* *pa1 같은* 경우입니다.txt는 0을 받습니다.

# 컴파일링

다음 명령을 사용하여 Java 코드를 테스트할 수 있습니다.

%./gradlew -qrunTestRunner

다음 명령을 사용하여 제출용 zip 파일을 만들 수도 있습니다. 학생 ID(.txt 파일 이름)로 명명된 zip 파일은 "build" 디렉토리에 있습니다.

%./gradlew -q zip 제출

표준 입출력을 사용하여 테스트할 수 있는 빈 Main 클래스를 제공합니다.

%./gradlew -q --laps=slaps runMain

이 파일(src/main/Main.java)은 제출용이 아니기 때문에 파일의 패키지를 사용할 수 있습니다.

윈도우즈의 경우 ./gradlew 대신 gradlew.bat를 사용합니다.

# 테스트

코드 확인을 위한 소규모 테스트 스위트(src/test)를 제공했습니다. 테스터 프로그램을 컴파일하여 실행하면 코드를 테스트할 수 있습니다.

코드 등급을 매기는 데 사용할 테스트 제품군은 여기에 제공된 테스트 제품군보다 훨씬 더 엄격합니다(제공된 테스트의 상위 집합은 아님). 코드를 좀 더 철저하게 확인하기 위해 자신만의 테스트 케이스를 만드는 것을 고려해야 합니다.